VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Univerzální webový kalendář pro správu událostí v ASP.NET

Universal web calendar in ASP.NET

2009 Pavel Starzyczny

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 5. května 2009 .....

Rád bych na tomto místě poděkoval svému vedoucímu bakalářské práce Ing. Michalu Radeckému za trpělivost, odborné vedení a spoustu užitečných rad.

Abstrakt

Kalendář je užitečnou pomůckou pro naše každodenní aktivity, který nám dává jasný přehled nad naším prožitým časem. Tato bakalářská práce si klade za cíl prozkoumat existující řešení v oblasti elektronického plánování času a na základě těchto poznatků navrhnout vhodný univerzální systém pro prezentaci naplánovaných událostí svým uživatelům. Práce obsahuje analýzu, návrh a implementaci takovéhoto univerzálního kalendáře, který bude postaven na technologii ASP.NET.

Klíčová slova: ASP.NET, kalendář, prezentace událostí, assembly

Abstract

Calendar is an useful tool for our every day activities because it gives us a clear overview about our passed time. This bachelor thesis sets a task to explore existing electronic schedule solutions and following these information propose an universal system of displaying the scheduled events to its user. This work consists of an analysis, a layout and an implementation of the universal calendar which will be based on ASP.NET technology.

Keywords: ASP.NET, calendar, events presentation, assembly

Seznam použitých zkratek a symbolů

AJAX	–	Asynchronous JavaScript + XML
ASP.NET	–	Active Server Pages na platformě .NET
C#	–	Programovací jazyk C#
CSS	–	Cascading Style Sheets
DOM	–	Document Object Model
HTML	–	HyperText Markup Language
iCal	–	iCalendar
JavaScript	–	JavaScript
MS	–	Microsoft
SQL	–	Structured Query Language
W3C	–	World Wide Web Consortium
WSDL	–	Web Services Description Language
XML	–	eXtensible Markup Language

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Již od pradávna má člověk tendenci zaznamenávat události, které se kolem něj dějí nebo které ho teprve čekají. Proto vznikala potřeba tyto události mít někde zapsány. Lidé si začali události zapisovat na jedno místo, aby měli přehled o využitém či volném čase. Jako nejdůležitější informací, která je o určité události zaznamenaná, je datum a čas, kdy tato událost nastala či teprve nastane. Proto je více než vhodné zobrazení událostí společně s kalendářem. Toto spojení dává možnost zobrazit člověku události v přehledné formě. V dnešní době máme mnoho možností zápisu. A to v psané či elektronické podobě.

Výhodou ručně zaznamenávaných událostí je rychlost jejich zapsání třeba na papír, nebo do diáře. Nepotřebujeme k tomu žádnou elektrickou energii ani zařízení, jako notebook, tablet nebo mobilní telefon. Nevýhodou je ale například nemožnost na tyto události upozorňovat a také třeba jen i obyčejná spotřeba papíru.

Efektivnějším způsobem je ukládat záznamy o událostech v elektronické podobě. A to tak, abychom k nim měli přístup, pokud možno odkudkoli a neustále. Na základě těchto požadavků vzniklo mnoho různých programových řešení, které nabízejí nepřeberné množství nastavení a možností, jak s událostmi pracovat. Tato řešení mohou být lokální nebo vzdálená.

První možností jsou programy běžící lokálně na osobním počítači, mobilu či PDA. Tato řešení jsou vhodná pro jednotlivce, kteří chtějí využívat komfort programu, nainstalovaného na svém počítači či mobilním telefonu. Avšak nevýhodou je nemožnost prezentovat a sdílet události jinak, než na lokálním počítači.

Druhou možností jsou řešení, která běží vzdáleně, na nějakém serveru a je k nim přístup například prostřednictvím Internetu. Tato řešení mají výhodu v dostupnosti informací pro mnoho nezávislých uživatelů, kteří nemusí instalovat jakékoli dodatečné programy. Pro zobrazení informací jim postačí například internetový prohlížeč.

Samozřejmostí je, že existují také řešení, která kombinují výhody lokálních a vzdálených řešení a propojují je do jednoho celku.

2 Zmapování existujících řešení

Jak již bylo uvedeno v úvodu, události je možno prezentovat lokálně, nebo vzdáleně. Pro porovnání jednotlivých řešení byl vybrán vždy jeden zástupce z obou skupin. Tito zástupci budou podrobeni hlubšímu zkoumání svých funkcí.

2.1 Google Kalendář

Jako zástupce vzdálených řešení byla vybrána aplikace od společnosti Google, a to Google Kalendář. Tato aplikace je dostupná na internetových stránkách www.google.com/calendar. Pro vstup do aplikace je potřeba mít zřízen emailový účet u společnosti Google a pod tímto účtem se také přihlásit.

2.1.1 Pracovní prostředí aplikace

Jak je vidět na obrázku 1, po přihlášení je na levé straně grafické plochy zobrazen kalendář s aktuálním měsícem a v něm zvýrazněnými dny, ve kterých jsou události vloženy. Na pravé straně jsou pak vyznačeny události aktuálního týdne. Události se mohou zobrazit pro daný den, týden, měsíc nebo pro čtyři dny. Tímto si uživatel kalendáře může přehledně vyvolat právě tolik informací, kolik si přeje. Interakce s aplikací je bez nutnosti načítání stránek, což uživateli poskytuje dojem, jako by pracoval s aplikací na lokálním počítači.

Obrázek 1: Google Kalendář – grafické rozhraní

Aplikace nabízí zřízení více kalendářů. Díky tomu je možné události dělit tematicky, například na události soukromé a pracovní, což přispívá k přehlednosti. Další možnost je přidat si kalendáře jiných uživatelů. Tímto je umožněno zobrazit si události, které si ukládají vaši přátelé do svých kalendářů.

2.1.2 Správa událostí

Vkládání událostí se děje pomocí odkazu Vytvořit událost, nebo po kliknutí na grafickou plochu do určitého dne a hodiny. Jako základní informace o události je možno zadat:

- datum od kdy do kdy je událost platná
- jméno události
- místo konání
- text událost
- do kterého kalendáře událost patří

Dalším dostupným nastavení je opakování. Zde se nabízí výběr z osmi variant. U každé události je možnost nechat si poslat upozornění na svou emailovou adresu a nastavit, jak dlouho před událostí se má toto upozornění odeslat. Kalendář umožňuje nastavit úroveň ochrany osobních údajů události buďto jako soukromou nebo veřejnou.

Po vytvoření nové události je událost zobrazena na grafické ploše a je možno ji dále editovat, mazat či přesouvat do dalších dnů pomocí myši a posunu po grafické ploše.

2.1.3 Prezentace událostí na dálku

Další funkcí nepřímo spojenou s událostí je možnost vystavit zvolený kalendář na své osobní stránky nebo blog. K tomuto vám aplikace vygeneruje jedinečný kód, který se umístí do internetové prezentace jako vložené internetové okno `iframe`.

```
<iframe src="http :// www.google.com/calendar/embed?src=jmeno%40gmail.com&ctz=Europe/
    Prague" style="border:0" width="800" height="600" frameborder="0" scrolling="no"></iframe>
```

Výpis 1: Ukázka vložení kalendáře do HTML stránky

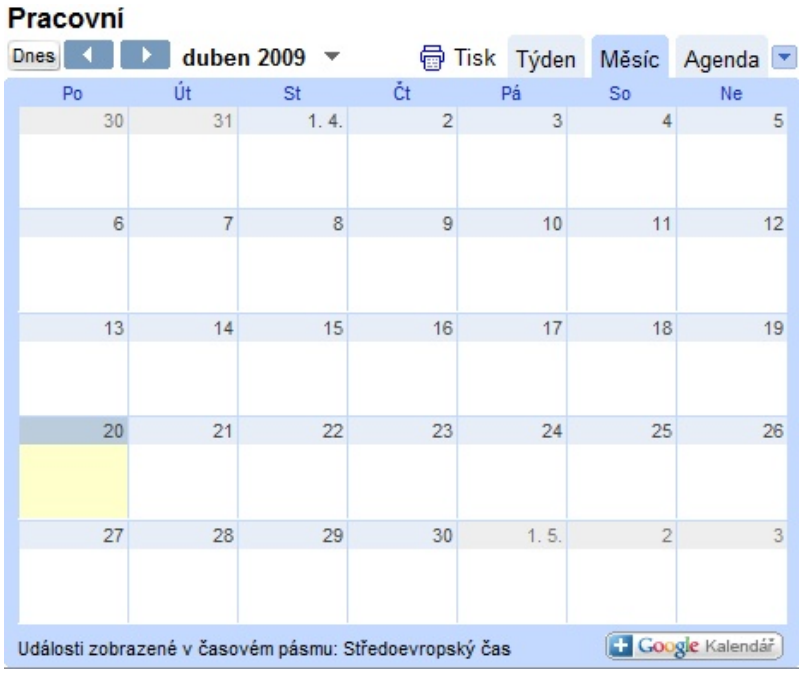
Také je možnost si uložit pouze internetový odkaz na kalendář, který je uveden v atributu `src` výpisu 1. Pak se zobrazí vybraný kalendář s událostmi, jak je vidět na obrázku 2. Volitelně je uživateli umožněno měnit barevné schéma kalendáře.

Na příkladu je také vidět, že veškerá logika aplikace, jak pracovní prostředí, tak i prezentace událostí, je umístěna na jednom místě, a to na serverech společnosti Google. Všechny informace o kalendáři a událostech v něm obsažených se získávají přes unikátní URL adresu.

Pro zobrazení událostí i v jiných prostředích poskytuje Google Kalendář kromě URL adresy ještě export vybraného kalendáře do formátů XMl a iCal.

Export do formátu XML slouží jako zdroj dat pro zobrazení událostí v libovolné čtečce XML zdrojů.

Druhým formátem je pak iCal (též iCalendar). Tento formát je standart pro výměnu kalendářových dat a je podporován velkým množstvím programových řešení. iCalendar



Obrázek 2: Google Kalendář – zobrazení samotného kalendáře

umožňuje uživatelům posílat žádosti k naplánování schůzky nebo úkolu ostatním uživatelům prostřednictvím standardního emailu. Příjemce emailové zprávy s takovýmto obsahem může jednoduše odesilateli odpovědět stejnou cestou (podmínkou je programové vybavení pro zpracování těchto dat), včetně možnosti úpravy navržených parametrů schůzky resp. úkolu. Data iCalendar jsou typicky používána ve spojení s emailovou komunikací, standard je ovšem navržen tak, aby byl nezávislý na přenosovém protokolu.[3]

2.2 Rainlendar Lite

Jako zástupce aplikací běžící na lokálním počítači byl vybrán program Rainlendar Lite dostupný na adrese http://www.rainlendar.net. Ve verzi Lite je tato aplikace ke stažení jako freeware.

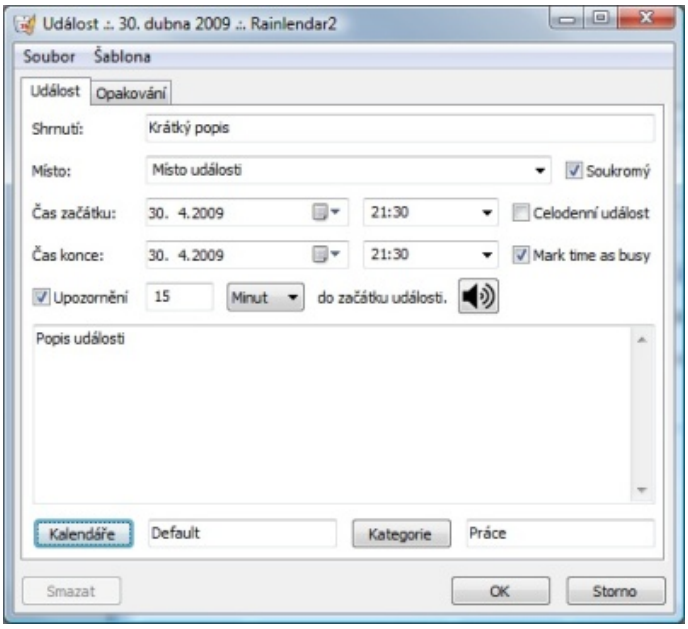
2.2.1 Pracovní prostředí aplikace

Po dokončení instalace aplikace je na pracovní ploše počítače zobrazen kalendář s vyznačeným aktuálním dnem, jak je vidět na obrázku 3. Celé pracovní prostředí aplikace je tvořeno tímto kalendářem.

V tomto kalendáři můžeme plynule, pomocí šipek, přecházet mezi jednotlivými měsíci. Při přejetí myši nad dnem s událostmi se zobrazí jejich podrobnosti. Toto zobrazení je interaktivní tak, jak dovolují aplikace instalované na lokálním počítači, čili bez přeblikávání, jako tomu je u některých řešení na internetu.

Obrázek 3: Rainlendar – zobrazení kalendáře

Při vytvoření události, jak je vidět na obrázku 4, je možno zadat informace o události, zařadit událost do různých kalendářů a také do různých kategorií. Dále je pak možnost na upozornění na událost a nebo opakování události.



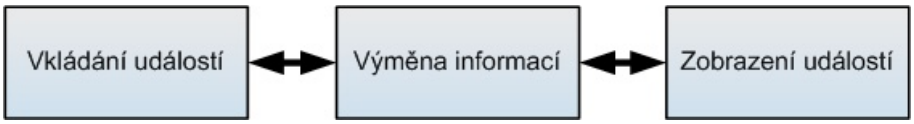
Obrázek 4: Rainlendar – vytvoření události

2.2.2 Prezentace událostí

Nevýhodou prezentace událostí, jak bylo zmíněno v úvodu, je nemožnost interaktivně sdílet události s jinými uživateli. Tento nedostatek částečně odstraňuje možnost exportovat vložené události do formátu iCal, jak tomu bylo i u aplikace Google Kalendář. Touto možností ale není dosaženo prezentace událostí v reálném čase.

3 Obecná charakteristika kalendáře

Po prozkoumání existujících řešení a vyhodnocení jejich funkcionality a přínosu pro uživatele vyplynuly následující požadavky na Kalendář. Jelikož má být kalendář vytvářen jako webová aplikace a ne jako lokální, pak jako vzorový příklad takovéto aplikace může posloužit řešení kalendáře od společnosti Google. Je v něm obsaženo mnoho zajímavých postřehů a funkcí, které by mohly být převzaty a zakomponovány i do právě vytvářené aplikace. Dále je výhodné rozdělit funkčnost celé aplikace na několik samostatných částí, kde každá část by měla svou funkcionalitu nezávislou na jiné. Navenek by pak ale vytvořily jeden funkční celek. Pro takovouto funkcionalitu bude nutné navrhnout části, které jsou vidět na obrázku 5.



Obrázek 5: Obecná spolupráce vrstev

První část by se měla starat o zanášení údajů do databáze. Část druhá pak o jejich zobrazování. Mezi těmito částmi bude figurovat část třetí, která bude získávat uložené události a ty dále poskytovat prvku zobrazujícímu události.

3.1 Vkládání událostí

Kalendář bude vytvořen jako víceuživatelská aplikace. V tomto případě je nutné zajistit registraci a přihlašování uživatelů. Důležité je, aby se uživatelé mezi sebou nemohli nijak ovlivňovat a nemohli používat kalendáře jiných uživatelů bez jejich vědomí. Toto také znamená, že aplikace nesmí mít uživatele s právy takovými, aby mohl jakkoli zasahovat do kalendářů jiných uživatelů.

3.1.1 Správa kalendářů

Pokud je uživatel v systému nový, nemá vytvořen žádný kalendář. Prvním krokem tedy bude jeho založení . Nový kalendář by měl obsahovat tyto informace:

- jméno
- popis
- zda je kalendář veřejný či soukromý
- uživatele, oprávněné pro práci s kalendářem

Aplikace uživateli umožní vytvoření více kalendářů, do kterých si bude vkládat své události. Tato možnost částečně odstraní nutnost používání řazení událostí do kategorií, jak tomu bylo u aplikace Rainlendar. Také bude možno povolit jinému uživateli přístup ke

svému kalendáři. Tímto povolením dá jeden uživatel druhému možnost vkládat, mazat a editovat události v tomto kalendáři obsažené, ale nebude moci smazat či editovat informace o tomto kalendáři. Tyto role uživatelů se dají popsat jako přispěvatel a redaktor.

3.1.2 Správa událostí

Základním kamenem aplikace bude vkládání a správa událostí. Jak již bylo řečeno, tyto události uživatelé mohou vkládat do svých kalendářů nebo do kalendářů, které jim byly přiděleny. Události by měly povinně obsahovat následující informace:

- jméno události
- datum a čas začátku
- datum a čas konce

Události budou moci být vícedenní. Čas začátku a konce je možné vybrat v půlhodinovém intervalu, což znamená, že událost může začít nejdříve o půlnoci tj. 00:00 a pokračovat po půlhodině až do 23:30 téhož dne nebo až do půlnoci dne jiného.

Pouze tyto informace budou dostačující pro založení nové události, jelikož uživatel nemusí mít potřebu nijak více specifikovat událost, kterou chce uložit. Toto může udělat později nebo také vůbec.

Dalšími nepovinnými možnostmi, by měly být:

- krátký popis události
- text událost
- místo konání (například Vysoká škola Báňská atd.)
- místnost
- adresa
- internetový odkaz na mapu místa konání
- přiložený soubor

Další funkcí je zobrazování událostí podle týdne, měsíce a dne. Protože tato funkce je velice užitečná a uživatelům zpřehlední náhled událostí, bude také začleněna.

3.2 Zobrazování událostí

Zobrazení událostí kalendáře by mělo fungovat kdekoli na internetu. Rozdíl mezi tímto řešením a řešením od firmy Google bude spočívat v tom, že k tomuto účelu bude využita univerzální komponenta, která nebude fyzicky umístěna na místě, kde se události

vkládají. Toto je podstatný rozdíl proti Google Kalendáři, který toto řešení má implementované pouze jako internetový odkaz na svůj server, který vše provádí na jednom místě.

Od této komponenty se očekává, že nebude fyzicky svázána s místem, kde se události vkládají. Hlavní a jedinou funkcí bude zobrazování události z uživatelem zadaného kalendáře. Komponenta by měla být co nejvíce přizpůsobitelná požadavkům uživatele. V důsledku to znamená, že vše, co komponenta zobrazí, musí být nějakým způsobem nastavitelné.

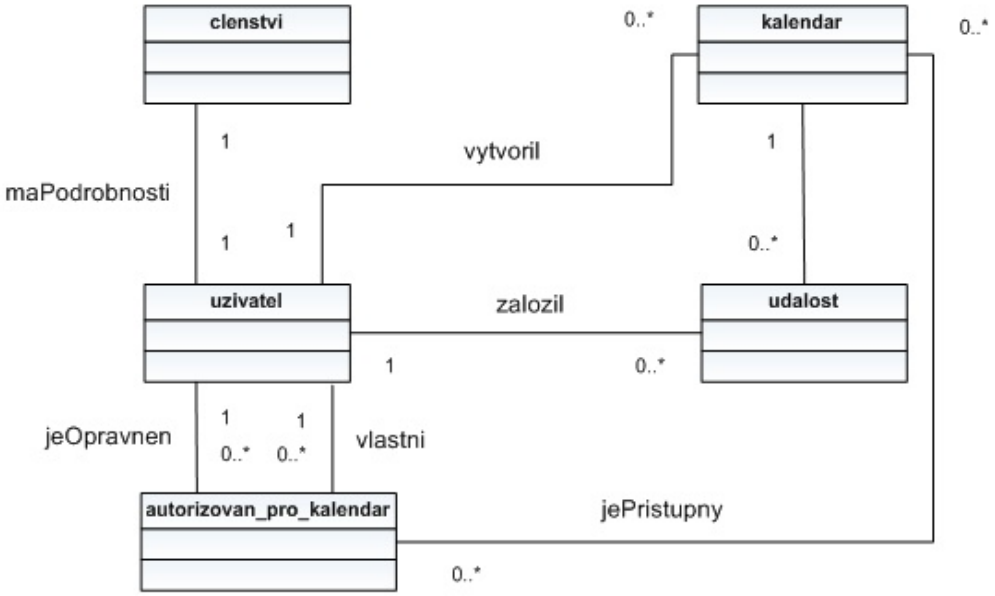
3.3 Výměna informací

Jelikož prezentační část aplikace bude nezávislá na administrační části, bude nutno vytvořit službu, která bude zprostředkovávat interakci mezi těmito dvěma vrstvami. Tato služba by měla běžet na stejném místě jako administrační část.

4 Analýza systému

4.1 Datové prostředí

Třídní diagram na obrázku 6, definuje základ statického popisu administrační části. Diagram odhaluje jaké objekty, a tím i třídy, potřebuje k realizaci požadované funkcionality, v jakých jsou tyto třídy relacích a které zprávy jsou mezi instancemi těchto tříd předávány.[4]



Obrázek 6: Třídní diagram

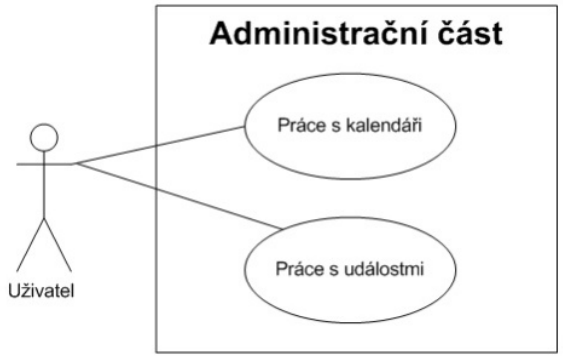
V příloze A je uveden datový slovník, který obsahuje seznam datových objektů v databázi a obsahuje údaje o jejich jménu a popisu, datovém typu, integritních omezeních, jestli jsou primárním nebo cizím klíčem, nebo jestli musí obsahovat nějakou hodnotu.[5]

4.2 Práce uživatelů se systémem

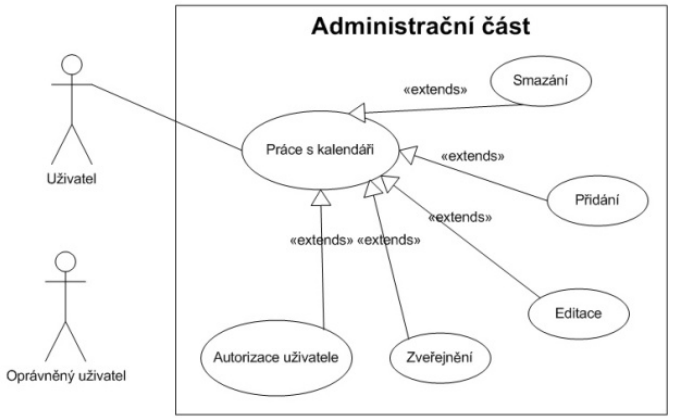
Obrázky 7, 8 a 9 ukazují práci uživatelů s administrační částí.

4.3 Chování systému

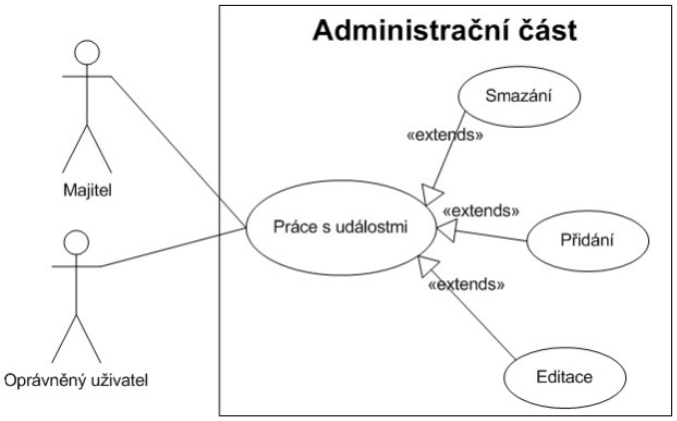
Diagram aktivit na obrázku 10 popisuje podnikový proces pomocí jeho stavů reprezentovaných vykonáváním aktivit a pomocí přechodů mezi těmito stavy způsobených ukončením těchto aktivit. Účelem diagramu aktivit je blíže popsat tok činností daný vnitřním mechanismem jejich provádění.[4]



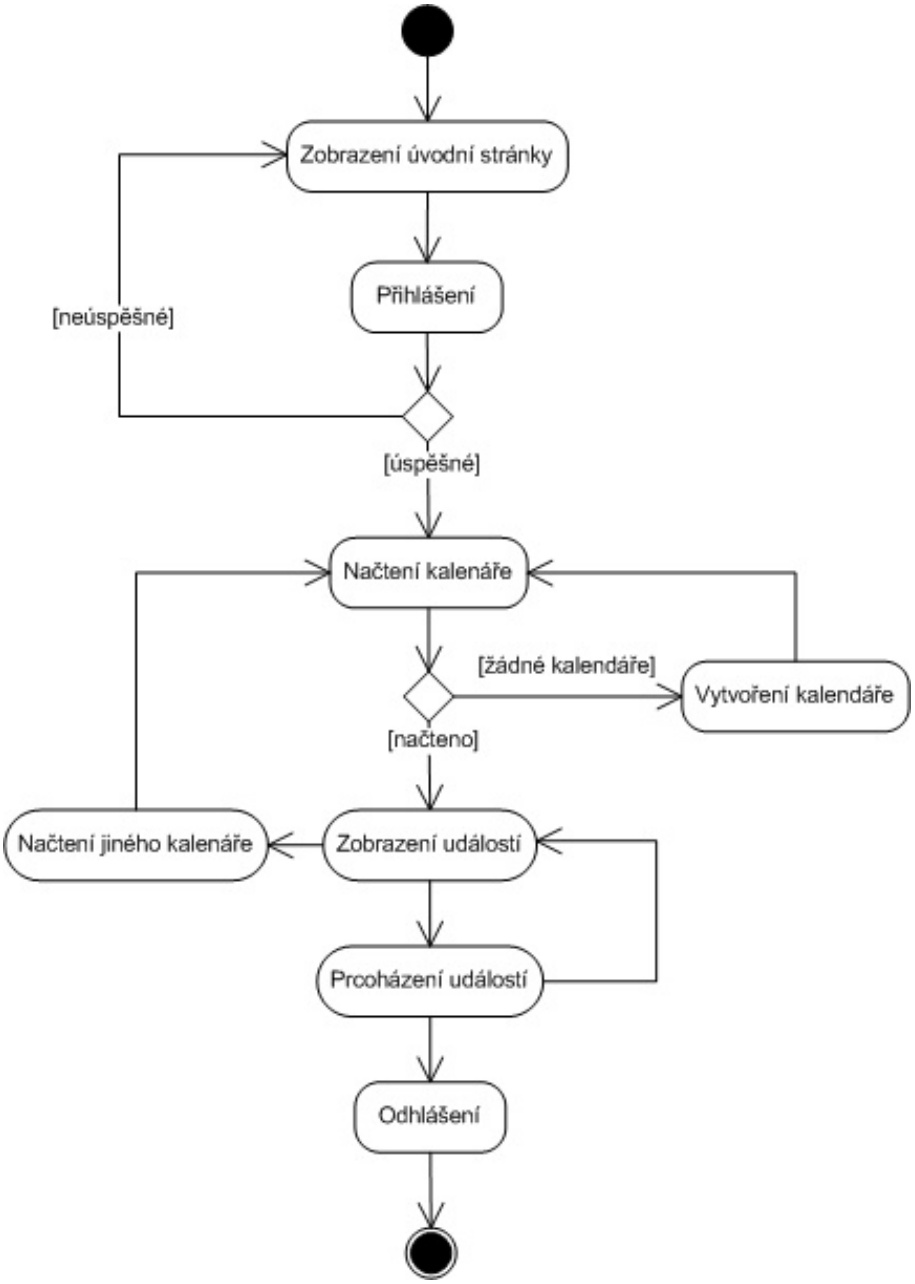
Obrázek 7: Základní práce uživatele



Obrázek 8: Práce s kalendáři



Obrázek 9: Práce s událostmi



Obrázek 10: Aktivitní diagram práce s administrační částí

5 Specifikace a návrh systému

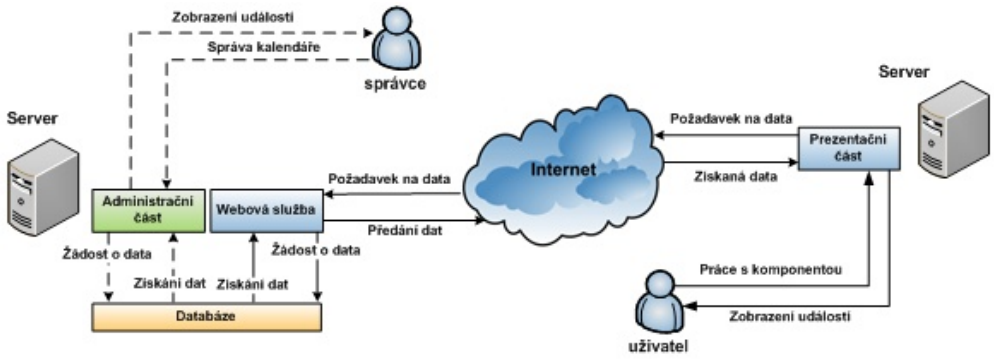
5.1 Celkový koncept

Jak již bylo zmíněno v Obecné charakteristice kalendáře [3/10], celkový koncept aplikace bude rozdělen na následující části:

1. administrační část
2. prezentační část
3. službu výměny informací

Všechny části budou na sobě nezávislé a jedna druhou nebude moct nijak ovlivnit. Jak je vidět na obrázku 11, uživatel komunikuje s administrační částí a ta získává data z databáze a předá je zpátky uživateli.

Prezentační část komunikuje skrze Internet s webovou službou, která podle informací zaslaných prezentační částí získá data z databáze. Získaná data poté vrátí zpět a prezentační část je následně zobrazí uživateli.



Obrázek 11: Schéma komunikace

5.2 Administrační část

Administrační část umožňuje kompletní správu událostí a kalendářů a jejich nastavení. Aplikace je volně dostupná na internetu a je do ní umožněn přístup každému, kdo projde registračním procesem a úspěšně se přihlásí do systému. Z tohoto plyne, že systém neobsahuje uživatele s právy pro zobrazení všech kalendářů a událostí všech registrovaných uživatelů. Toto omezení je zavedeno z bezpečnostních důvodů.

Po přihlášení do administrace nemá nově zaregistrovaný uživatel žádné předem vytvořené kalendáře.

Uživatel může s kalendářem pracovat těmito způsoby:

- vytvářet, editovat a smazat své kalendáře

- vytvářet, editovat a mazat události ve svých kalendářích
- přidávat a odebírat oprávněné uživatele pro práci s kalendářem
- vytvářet, editovat a mazat události v kalendářích, ve kterých je uveden jako oprávněný uživatel
- nastavovat kalendář jako veřejný či neveřejný
- zobrazovat události pro daný den, týden a měsíc
- k událostem nahrávat soubory

Co je uživateli zakázáno:

- editovat a smazat kalendáře, ve kterých je uveden jako oprávněný uživatel
- zobrazit kalendáře ostatních uživatelů systému, pokud v nich není uveden jako oprávněný uživatel

5.3 Prezentační část

Po zohlednění všech funkcí a požadavků na komponentu bylo rozhodnuto, že základní funkcionalitu poskytne zabudovaná komponenta od společnosti Microsoft - Kalendář ASP.NET (Calendar ASP.NET). Tato komponenta je dostupná, jako mnoho dalších, při vývoji stránek na platformě .Net. Výhodou této komponenty je její propracovanost. Základní funkce této komponenty, bez žádného dalšího nastavování, je zobrazení aktuálního měsíce a vyznačení aktuálního data v měsíci.

Při tvorbě komponenty připadaly v úvahu dvě možnosti. Vytvořit kalendář jako vlastní serverový ovládací prvek nebo jako uživatelský ovládací prvek. Aby mohlo být posouzeno, která technika je nejlepší pro vytvoření komponenty, je třeba vědět, jaké jsou možnosti těchto ovládacích prvků.

5.3.1 Uživatelské ovládací prvky

Uživatelské ovládací prvky (user controls) je malá část stránky, která může obsahovat statický kód HTML a webové ovládací prvky. Předností uživatelského ovládacího prvku je, že jakmile jej vytvoříte, dá se opětovně použit na jakémkoli místě v téže webové aplikaci. K tomuto ovládacímu prvku můžete přidávat vlastní vlastnosti, události a metody. Uživatelské ovládací prvky se tedy hodí například na tvorbu jednotných formulářů, záhlaví a zápatí stránky nebo také pokud potřebujete standardizovat menu na všech svých webových stránkách. Soubory uživatelských ovládacích prvků jsou obdobou souborů webových formulářů ASP.NET, avšak pro soubory používají příponu `.ascx`. Můžeme v nich použít přímé in-line skripty nebo kód v pozadí (code behind). [1]

5.3.2 Serverové ovládací prvky

Na rozdíl od uživatelských ovládacích prvků jsou serverové ovládací prvky (server controls) z pohledu vývojáře složitější na vytvoření, zato ale máme úplnou kontrolu nad HTML kódem, který vygenerujeme. Čili může vytvořit vlastní serverový prvek, který má úplně stejné vlastnosti jako například Kalendář ASP.NET a rozšířit jeho funkčnost. Serverové ovládací prvky můžeme také umístit do panelu Toolbox Visual Studia, nastavovat jejich vlastnosti a události v návrhovém režimu. Navíc všechny webové ovládací prvky dostupné na platformě .NET jsou serverovými ovládacími prvky.[1]

Z těchto vlastností jasně vyplynulo, že námaha na vytvoření vlastního serverového ovládacího prvku se vyplatí a přinese své ovoce. Proto jako základ kalendářové komponenty poslouží webový ovládací prvek Kalendář ASP.NET, který bude rozšířen o další funkcionalitu.

5.4 Použité technologie

Pro implementaci aplikace byly použity jazyky ASP.NET a C# na platformě Microsoft .NET verze 3.5. Data jsou uložena v databázi Microsoft SQL Server 2005. Pro zobrazování událostí na straně klienta je použita technologie Javascript.

5.4.1 ASP.NET

ASP.NET je technologie společnosti Microsoft a je odvozena od starší technologie pro vývoj webů ASP. Obě technologie jsou velmi odlišné. ASP.NET je založen na CLR (Common Language Runtime), který je sdílen všemi aplikacemi postavenými na .NET Frameworku. Programátoři tak mohou realizovat své projekty v jakémkoliv jazyce podporujícím CLR, např. Visual Basic.NET, JScript.NET, C#, ale i mutace Perlu, Pythonu a další. Aplikace založené na ASP.NET jsou také rychlejší, neboť jsou předkompilovány do jednoho či několika málo DLL souborů, na rozdíl od ryze skriptovacích jazyků, kde jsou stránky při každém přístupu znovu a znovu zpracovávány.

Koncept ASP.NET WebForms ulehčuje programátorům přechod od programování klasických aplikací pro Windows do prostředí webu: stránky jsou poskládány z objektů, ovládacích prvků (Controls), které jsou protějškem ovládacích prvků ve Windows. Při tvorbě webových stránek je tedy možné používat ovládací prvky jako tlačítko, nápis a další. Těmto prvkům lze přiřazovat určité vlastnosti, zachytávat na nich události, atd. Tak, jako se ovládací prvky pro Windows samy kreslí do formulářů na obrazovku, webové ovládací prvky produkují HTML kód, který tvoří část výsledné stránky poslané do klientova prohlížeče.[3]

5.4.2 ASP.NET AJAX Control Toolkit

ASP.NET AJAX Control Toolkit je rozšíření klasického ASP.NET. Obsahuje množství interaktivních serverových ovládacích prvků využívající technologie AJAX, která obchází některá omezení jazyka HTML. Výhodou používání těchto komponent je, že vývojář nemusí vědět, jak ovládací prvek funguje a jak je stavěn, ale pouze využívá funkcionality,

kterou ovládací prvek poskytuje. S těmito dodatečnými komponentami se pracuje úplně stejně, jako se základní sadou komponent dostupných na platformě .NET.[3]

5.4.3 MS SQL Server

MS SQL Server je relační databázový systém vyvinutý společností Microsoft. Jeho hlavním dotazovacím jazykem je SQL, který se používá pro práci s daty.[3]

5.4.4 Webové služby

Webová služba je způsob, jak spolu mohou aplikace komunikovat vzdáleně prostřednictvím Internetu. Pomocí webových služeb spolu mohou komunikovat aplikace psané v odlišných jazycích a běžící na různých platformách. Ke komunikaci mezi aplikacemi se používá protokol SOAP (Simple Object Access Protocol), což je standart pro výměnu zpráv založený na formátu XML standardizovaný organizaci W3C. Protokol SOAP je také nezávislý na transportním protokolu a implementaci klienta a serveru. Pro popis toho, jaké webová služba nabízí funkce a jak se na tyto funkce zeptat, slouží popisovací jazyk WSDL (Web Services Description Language), taktéž ve formátu XML.[3]

5.4.5 Javascript

Javascript je objektově orientovaný skriptovací jazyk, který je často vkládán přímo do HTML kódu. Jelikož je spouštěn na straně klienta a ne na straně serveru, jak tomu je například PHP či ASP, plynou pro něj jistá omezení. Například není možné pracovat se soubory, aby nebylo porušeno soukromí uživatele. Javascript je hojně využíván pro dynamickou práci s obsahem HTML stránky. Umožňuje dynamicky měnit obsah HTML prvků, vykonávat cykly a podmínky, jako je tomu u jiných jazyků a mnoho dalšího.[3]

5.4.6 DOM

DOM je objektově orientovaná reprezentace XML nebo HTML dokumentu. DOM je aplikační prostředí umožňující přístup či modifikaci obsahu, struktury nebo stylu dokumentu, či jeho částí. Původně měl každý webový prohlížeč své vlastní specifické rozhraní k manipulaci s HTML elementy pomocí JavaScriptu. Vzájemná nekompatibilita těchto rozhraní však přivedla W3C k myšlence standardizace, a tak vznikl W3C Document Object Model (zkráceně W3C DOM). Tato specifikace je platformně a jazykově nezávislá. Předchozí specifická rozhraní byla nazvána Intermediate DOM (anglicky přechodný DOM). DOM umožňuje přístup k dokumentu jako ke stromu podobnému adresářové struktuře operačního systému.[3]

5.4.7 XHTML a CSS

XHTML je značkovací jazyk pro tvorbu hypertextových dokumentů v prostředí WWW vyvinutý W3C. Je nástupcem staršího HTML verze 4.01, jehož vývoj byl touto verzí ukončen. XHTML splňuje požadavky na XML dokument a má tři verze:

1. XHTML 1.0 Strict
2. XHTML 1.0 Transitional
3. XHTML 1.0 Frameset

Odlišnosti těchto verzí jsou pouze v tom, jaké formátovací značky v nich můžete používat.

CSS byly vyvinuty pro osvobození HTML kódu od definice vzhledu dokumentu. Definice vzhledu by již měla být v odděleném souboru kaskádových stylů a vzhled by se neměl vpisovat do kódu samotného.[3]

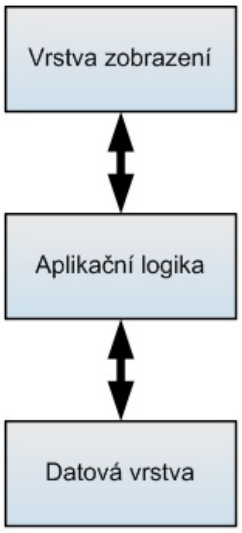
5.4.8 C#

Programovací jazyk C# je výkonný, ale přitom jednoduchý, komponentně orientovaný jazyk zaměřený především na vývojáře aplikací v prostředí .NET Framework. Zdědil velké množství toho nejlepšího z jazyků C++ a Microsoft Visual Basic, ale jen málo jejich nesrovnalostí, takže výsledkem je čistší a logičtější jazyk. [2]

6 Implementace systému kalendáře

6.1 Administrační část

Administrační část je napsána v jazyce ASP.NET a je programována jako třívrstvá aplikace, která je zobrazena na obrázku 12.



Obrázek 12: Schéma komunikace částí aplikační vrstvy

Vrstva zobrazení je tvořena soubory s příponou `.aspx`, ve kterých je definován základní vzhled stránky a definice všech prvků na stránce. Na ní navazuje vrstva aplikační logiky, která je umístěna v souborech s příponou `.cs` a která přebírá data od datové vrstvy a plní jimi prvky na jednotlivých stránkách. Datová vrstva je umístěna v samostatné assembly s příponou `.dll` a obsahuje funkce pracující s databází.

6.1.1 Struktura aplikace

Struktura aplikace je rozdělena do několika částí:

App_Code\ adresář s výkonným kódem webové služby

App_Data\ datové soubory a databáze

Bin\ dll assembly potřebné k chodu aplikace

Images\ adresář s obrázky

LoggedUser\ adresář se stránkami, ke kterým má přístup přihlášený uživatel

UserControls\ adresář s uživatelskými ovládacími prvky

UsersFiles\ adresář s nahranými soubory uživatelů

WebService\ adresář obsahující definici webové služby

web.config konfigurační soubor aplikace

ostatní soubory v kořenové složce aplikace soubory pro nepřihlášené uživatele

6.1.2 Programový kód

Pro zobrazení událostí v aplikační části je použit standardní webový ovládací prvek Kalendář ASP.NET (Calendar ASP.NET) rozšířený o funkci zvýraznění dnů, ve kterých jsou zavedeny události. Pro zobrazení jednotlivých událostí je využito vlastní uživatelské komponenty. Tento vlastní webový ovládací prvek je vhodný pro znovupoužitelnost kódu. Protože může být zobrazeno více událostí najednou, je tato funkcionalita výhodná. Při generování jednotlivých událostí se vytvoří nová instance vlastního webového uživatelského prvku a té se předají jiná data k zobrazení. V tomto ovládacím prvku je navíc použit dodatečný webový ovládací prvek `CollapsiblePanelExtender` z balíku komponent ASP.NET AJAX Control Toolkit . Tento ovládací prvek, výpis 2, umožňuje dynamicky animovat zobrazování a schovávání jakéhokoli prvku. Protože tento prvek není standardní součástí ASP.NET, je zde uvedena ukázka jeho nastavení.

```
<ajaxcontrol:CollapsiblePanelExtender
    ID="CollapsiblePanelExtender"
    runat="server"
    TargetControlID="ContentPanel"
    ExpandControlID="TitlePanel"
    CollapseControlID="TitlePanel"
    Collapsed="true"
    ImageControlID="CollapseImage"
    CollapsedImage="~/Images/collapseDown.png"
    ExpandedImage="~/Images/collapseUp.png"
    CollapsedSize="0"
    ExpandDirection="Vertical"
    ExpandedText="(méně...)"
    CollapsedText="(více ...) "
    TextLabelID="InfoLabel" />
```

Výpis 2: Ovládací prvek CollapsiblePanelExtender

TargetControlID cílový ovládací prvek, který se má animovat

ExpandControlID ovládací prvek, který bude sloužit jako titulek

CollapseControlID po kliknutí na tento ovládací prvek bude spuštěno zobrazení/skrytí cílového prvku

Collapsed nastavení , zda se má cílový prvek animovat

ImageControlID ovládací prvek, který bude zobrazovat aktuální stav zobrazení cílového ovládacího prvku (skryt nebo zobrazen)

CollapsedImage URL obrázku, který bude zobrazen, pokud bude cílový ovládací prvek zobrazen

ExpandedImage URL obrázku, který bude zobrazen, pokud bude cílový ovládací prvek skryt

CollapsedSize výchozí výška skrytého ovládacího prvku

ExpandDirection směr animace, může být `Vertical` nebo `Horizontal`

ExpandedTex text informačního textu při zobrazeném cílovém prvku

CollapsedText text informačního textu při skrytém cílovém prvku

TextLabelID webový prvek `Label`, v kterém se zobrazí `ExpandedText` a `CollapsedText`

6.2 Prezentační část

Jelikož je celá bakalářská práce psaná v jazyce ASP.NET, předpokládá se, že komponenta bude taktéž naprogramována v tomto jazyce. Komponenta bude prezentována jako zkompilovaná assembly, což zaručuje jednoduchost práce a přenositelnost komponenty. Z výše uvedeného vyplývá, že komponenta je použitelná pouze pro stránky psané v jazyce ASP.NET. Komponenta by měla být také minimalistická a rovněž plně přizpůsobitelná požadavkům implementátora, což v důsledku znamená, že na stránce se výsledný kalendář událostí musí zobrazit tak, aby zabíral co nejméně místa a vypadal přesně tak, jak si implementátor přeje.

Implementace komponenty je rozdělena do pěti ucelených kroků:

- programování komponenty
- získání událostí ze vzdáleného serveru
- zobrazení událostí
- individuální přizpůsobení vzhledu komponenty
- nasazení komponenty v praxi

V těchto krocích bude podrobněji popsán způsob vývoje, aby bylo jasné, jak je komponenta naprogramována a jak spolu jednotlivé částí souvisí.

6.2.1 Programový kód

Základním požadavkem pro serverový ovládací prvek je, že musí dědit, ať už přímo nebo nepřímo, ze `System.Web.UI.Controls`. Většina webových ovládacích prvků není přímo odvozena z této třídy, ale ze třídy `System.Web.UI.WebControls`, která jim dává dodatečnou podporu například pro nastavení barvy textu nebo barvy pozadí.[1]

Protože Kalendář ASP.NET již dědí ze třídy `System.Web.UI.WebControls`, vlastní komponenta může dědit přímo z Kalendáře ASP.NET.

Následuje zobecnělá ukázka základního kódu vlastního serverového ovládacího prvku:

```
namespace CalendarControl
{
    [DefaultProperty("Text")]
    [ToolboxData("<{0}:CalendarControl runat=server></{0}:CalendarControl>")]
        public class CalendarControl : System.Web.UI.WebControls.Calendar
        {
            [Bindable(true)]
            [Category("Appearance")]
            [DefaultValue("")]
            [Localizable(true)]
                public string Text
                {
                    get
                    {
                        String s = (String)ViewState["Text"];
                        return ((s == null) ? "[" + this.ID + "]" : s);
                    }
                    set
                    {
                        ViewState["Text"] = value;
                    }
                }

                protected override void Render(HtmlTextWriter writer)
                {
                    writer.Write("kód před vyrenderování vlastní komponenty");
                    base.Render(writer);
                    writer.Write("kód po vyrenderování vlastní komponenty");
                }
        }
}
```

Výpis 3: Základ vlastního serverového ovládacího prvku

Kód ve výpisu 3 se nemusí psát ručně. Jemu podobný vygeneruje program Visual Studio sám, pokud zvolíme vytvoření nového projektu podle šablony ASP.NET Server control. Výše uvedený kód je upraven pro základní nastavení vlastní serverové komponenty. Celá třída dědí ze jmenného prostoru `System.Web.UI.WebControls.Calendar`. Třídy komponenty budou umístěny ve jmenném prostoru `CalendarControl`, jehož název nám bude suplovat jméno assembly, na kterou bude v projektu uvedena reference, pokud bude kalendář použit.

Důležitou částí je příznak `ToolboxData`, v němž je uveden předpis kódu, kterým komponentu vložíme do webových stránek, jak je vidět ve výpisu 4.

```
<cal:CalendarControl ID="CalendarControl1" runat="server" Text="text"/>
```

Výpis 4: Definice vlastního ovládacího prvku

Základní a nejdůležitější metodou je metoda `Render(HtmlTextWriter writer)`, která zajistí, že se serverový ovládací prvek správně vygeneruje na HTML kód. Důležité je uvědomit si, že pokud se volá událost bázové třídy `base.Render(writer)`, vygeneruje se komponenta, ze které je děděno. V tomto případě kalendář. Proto je v ukázce kódu, výpis 3, uveden jednoduchý příklad, jak zapsat vlastní kód před samotný kalendář a také po vygenderov8ní kalendáře. Tímto způsobem můžeme vlastní komponentu obalit do HTLM tagu `div` a tím zajistit například poziciovatelnost výsledného kódu na stránce. Za zmínku stojí také veřejná vlastnost `public string Text`, která umožňuje nastavovat a získávat nastavení z vně komponenty. Například při vložení komponenty do stránek a nastavením vlastnosti `Text` je tato vlastnost přístupná a je možno s ní dále pracovat.

Jelikož je děděno z Kalendáře ASP.NET, má komponenta přístup ke všem metodám, vlastnostem a událostem děděného prvku. Nejdůležitější vlastností komponenty je vlastnost SelectedDay , která obsahuje datum právě vybraného dne. Pokud budeme chtít, aby komponenta na základní události reagovala jinak než standardně, je zapotřebí takové události přetížit (klíčové slovo `override`) a napsat kód vlastní. V komponentě jsou přetíženy následující metody:

Render() , která se stará a samotné vykreslení vlastní komponenty.

OnLoad(EventArgs e) , která je spuštěna jako první při inicializaci komponenty. V této metodě se při prvním načtení stránky (první inicializaci komponenty) navazuje spojení se vzdáleným serverem pro získaní událostí.

OnSelectionChanged() je spuštěna vždy, když uživatel klepne v kalendáři na jiný den, než právě vybraný. Tato metoda také navazuje spojení se vzdáleným serverem a získává události ze vzdáleného serveru.

OnVisibleMonthChanged(DateTime newDate,DateTime previousDate) , která jako i předchozí navazuje spojení se vzdáleným serverem a získává události ze vzdáleného serveru. Důvodem, proč jsou si předchozí 3 metody podobné, ale nejsou stejné, je ten, že reagují na jiné události kalendáře a také ten, že vždy získávají události pro jiné rozpětí datumů . Tím je zajištěno, že při každé události se načtou správná data událostí.

OnDayRender(TableCell cell, CalendarDay day) Tato událost je klíčová, jelikož se spustí pro každý jednotlivý den v kalendáři. Metoda má 2 parametry. Parametr `cell` v sobě nese informaci o jedné buňce tabulky. Parametr `day` v sobě zase nese informaci o dnu, který je právě generován. Díky těmto parametrům je možná úplná kontrola nad vzhledem každé buňky tabulky, do které je kalendář renderován a můžeme tak odlišit dny, ve kterých se nachází nějaká událost a ve kterých ne.

V těchto metodách se nesmí zapomenou na volání události bázové třídy (klíčové slovo `base`), jinak by se nevyvolala základní funkcionalita těchto událostí a prováděl by se pouze kód, který byl do těchto událostí přidán.

6.2.2 Získání dat

K získání dat je třeba znát několik faktů, aby byla zvolena správná možnost. Je třeba si ujasnit, že prezentační část musí být:

1. nezávislá na serveru s administrační částí
2. implementátor používající komponentu nemá databázi událostí umístěnou na svém serveru
3. komponenta může být umístěna kdekoli na internetu bez předem známé URL adresy, a také může být umístěna jen na lokálním počítači s přístupem na internet
4. databáze s administrační částí je umístěna na serveru, který je přístupný odkudkoli z internetu pod předem známou a neměnnou URL adresou

Z výše uvedených faktů vyplynul jako vhodný způsob využít funkcionality Webových služeb pro získávání dat ze vzdáleného serveru. Podpora pro webové služby je v programu Visual studio zahrnuta a reprezentuje ji soubor s příponou `.asmx`. Tato webová služba běží na serveru spolu s administrační částí.

Pro získání dat pomocí webové služby je třeba znát přesnou URL adresu až k souboru s příponou `.asmx`, na které webová služba běží. V programu Visual studio se webová služba přidá pomocí dialogového okna Add Web Reference v okně Solution Explorer, zadá se adresa webové služby a potvrdí se. Poté je možno využívat metody webové služby, jako by byly naprogramované lokálně. Metody webové služby poté předají data v podobě odpojené sady dat `DataTable`, se kterou se bude dále pracovat.

6.2.3 Zobrazení událostí

Zobrazování událostí je to nejdůležitější, co komponenta obsahuje, a proto musí být zobrazení propracované. Po mnoha pokusech, jak zobrazit události kalendáře, bylo rozhodnuto jako nejvhodnější řešení použít další vlastní serverovou komponentu. Tato komponenta pracuje se všemi událostmi jednoho dne a má za úkol vygenerovat vhodný kus HTML kódu pro zobrazení okna s událostmi. Tato serverová komponenta dědí z `System.Web.UI.WebControls.Panel`, což znamená, že bez jakéhokoli přídavného kódu se vygeneruje pouze HTML kód `<div></div>`. Komponenta dědí z tohoto jmenného prostoru proto, že webový ovládací prvek Panel se nakonec vygeneruje do blokového HTML prvku `div`, který je vhodný na polohování a obalení nějakého druhu souvislých informací na stránce.

Pro zobrazení událostí na straně klienta byla zvolena technologie Javascript. Tato technologie umí pracovat s objektovým modelem dokumentu DOM a měnit jeho vlastnosti a styly, také se spouští na straně klienta, což přispělo k jednoduchosti zobrazování událostí nesčetnou měrou. Proto se události jednoho dne obalí HTML prvkem `div` s jedinečným identifikátorem. Tento identifikátor jedinečně identifikuje každé okno s událostmi pro jeden den. Díky těmto identifikátorům je možno využít vlastnosti `onmouseover` a spustit příslušný kód, který zobrazí nebo skryje okno s událostmi. Tato vlastnost je

přiřazena pouze buňkám dnů, ve kterých jsou nějaké události. Po přejetí myší nad vyznačeným dnem se zobrazí hlavičky událostí pouze pro daný den. Pro zobrazení detailů jednotlivých událostí je použit obdobný princip a události `onmouseover` a `onmouseout`. Využitím této techniky bylo docíleno toho, že pro zobrazení událostí pro daný den není potřeba kontaktovat server. Všechny události daného měsíce jsou zapsány ve stránce a jsou zobrazovány pomocí již zmíněné technologie Javascript. Nutnost kontaktování serveru nastává až tehdy, když si uživatel nechá zobrazit jiný měsíc. Tento způsob šetří přenos dat ze serveru ke klientovi a zrychluje práci s vygenerovaným kalendářem.

6.2.4 Individuální přizpůsobení vzhledu

Jelikož bude komponenta využívána na různých webových stránkách, není možné zajistit jednotný vzhled po každou stránku. Z tohoto faktu vznikl požadavek na plnou přizpůsobitelnost kalendářové komponenty. Tohoto je docíleno definováním vzhledu jednotlivých částí dvěma způsoby:

1. vzhled samotného kalendáře
2. vzhled okna událostí

6.2.4.1 Nastavení vzhledu kalendáře První způsob je definice vzhledu samotného kalendáře. Tato stylizace se provádí jako u standardní webového prvku Kalendář ASP.NET definováním příslušných doplňujících prvků přímo ve webové stránce.

Následuje ukázkový příklad, výpis 5, standardního zobrazení kalendáře podle šablony vestavěné v programu Visual Studio. Šablona má název Simple. Samozřejmostí je mnoho dalších nastavení, které je možno použít pro definici stylu zobrazení.

```
<cal:CalendarControl ID="CustomCalendar"
    CalendarGuid="00000000-0000-0000-0000-000000000000"
    EventDayColor="LightBlue"
    PopupWindowsSide="Right"
    PopupWindowsWidth="400"
    runat="server" BackColor="White" BorderColor="\#999999" CellPadding="4" DayNameFormat="
        Shortest" Font-Names="Verdana" Font-Size="8pt" ForeColor="Black" Height="180px"
    Width="200px" SelectionMode="DayWeekMonth" SelectMonthText="&amp;gt;">
        <SelectedDayStyle BackColor="\#666666" Font-Bold="True" ForeColor="White" />
        <SelectorStyle BackColor="\#CCCCCC" />
        <WeekendDayStyle BackColor="\#FFFFCC" />
        <TodayDayStyle BackColor="\#CCCCCC" ForeColor="Black" />
        <OtherMonthDayStyle ForeColor="\#808080" />
        <NextPrevStyle VerticalAlign="Bottom" />
        <DayHeaderStyle BackColor="\#CCCCCC" Font-Bold="True" Font-Size="7pt" />
        <TitleStyle  BackColor="\#999999" BorderColor="Black" Font-Bold="True" />
</cal:CalendarControl>
```

Výpis 5: Ukázka nastavení vlastní komponenty

K základnímu nastavení je potřebné vyplnit dodatečné nastavení, které je nutné pro správné fungování kalendáře. Proto je nutné definovat ještě následující vlastnosti:

CalendarGuid jedinečné označení chtěného kalendáře v administrační části systému

EventDayColor barva, kterou se označí dny s událostmi

PopupWindowsSide strana, na které se zobrazí okno s událostmi, možnostmi jsou `Left` nebo `Right`

PopupWindowsWidth šířka okna s událostmi v pixelech

6.2.4.2 Nastavení vzhledu okna událostí Druhým způsobem je stylování samotného okna s událostmi pomocí kaskádových stylů CSS. Okno událostí je vytvořeno pro co možná největší přizpůsobení potřebám vývojáře. Jedinou podmínkou pro vývojáře je, že musí do stránky připojit soubor kaskádových stylů. Poté je mu umožněno volně přizpůsobovat vzhled okna s událostmi.

Následuje popis CSS vlastností, kterými je možno vzhled přizpůsobit.

.calendar_popupWindow {...} obaluje okno s událostmi dne

.calendar_event {...} obaluje jednu událost

.calendar_eventDates {...} obaluje datum dne, pro které se události zobrazují

.calendarRow {...} definice pro každý řádek události

.calendar_hideButton {...} definice tlačítka pro zavření okna událostí

.calendar_hideButton img{...} obrázek pro zavření okna s událostmi

.calendar_eventName {...} obaluje jméno událost

.calendar_eventDescription {...} obaluje řádek krátký popis události

.calendar_eventText {...} obaluje řádek textu události

.calendar_eventDate {...} obaluje řádek data události

.calendar_eventAddress {...} obaluje řádek s adresou události

.calendar_eventAddressIn {...} definuje řádek adresy události

.calendar_eventRoom {...} obaluje řádek místnosti konání

.calendar_eventMap {...} obaluje řádek mapy události

.calendar_eventFile {...} obaluje řádek přiloženého souboru události

.calendar_clearBr {...} pro potřeby správného zarovnání a zalomení

6.2.5 Nasazení komponenty

Pro nasazení komponenty na jakýchkoliv stránkách je potřeba zkompilovaná DLL assembly `CalendarControl.dll` a soubor kaskádových stylů `calendarStyle.css`.

Pokud je kalendář použit, je třeba do projektu přidat referenci na DLL assembly komponenty a také začlenit kaskádový styl `calendarStyle.css`. Další důležitou podmínkou pro fungování kalendáře je předat komponentě správný identifikátor kalendáře, ze kterého se budou události zobrazovat. Tento identifikátor je uveden v nastavení kalendáře v administrační části a je ve tvaru 00000000-0000-0000-0000-000000000000 (nuly nahrazují jakoukoli číslici a znak z hexadecimálního rozsahu). Ten se musí předat jako parametr s názvem `CalendarGuid` kalendáři, ve kterém se tyto události zobrazí. Navíc musí být požadovaný kalendář nastaven v administrační části jako veřejný. Pokud tomu tak není, události se vůbec nestáhnou a kalendářová komponenta oznámí, že nemůže zobrazit události, protože to majitel kalendáře zakázal. Pokud se stane, že komponenta nebude schopna komunikovat se vzdáleným serverem, oznámí nedostupnost vzdáleného serveru.

6.3 Výsledná aplikace

Výsledkem předchozích částí je aplikace pro správu událostí a prezentační část ve formě samostatné komponenty, které splňují všechny výše uvedené požadavky na funkčnost a univerzálnost.

6.3.1 Administrační část

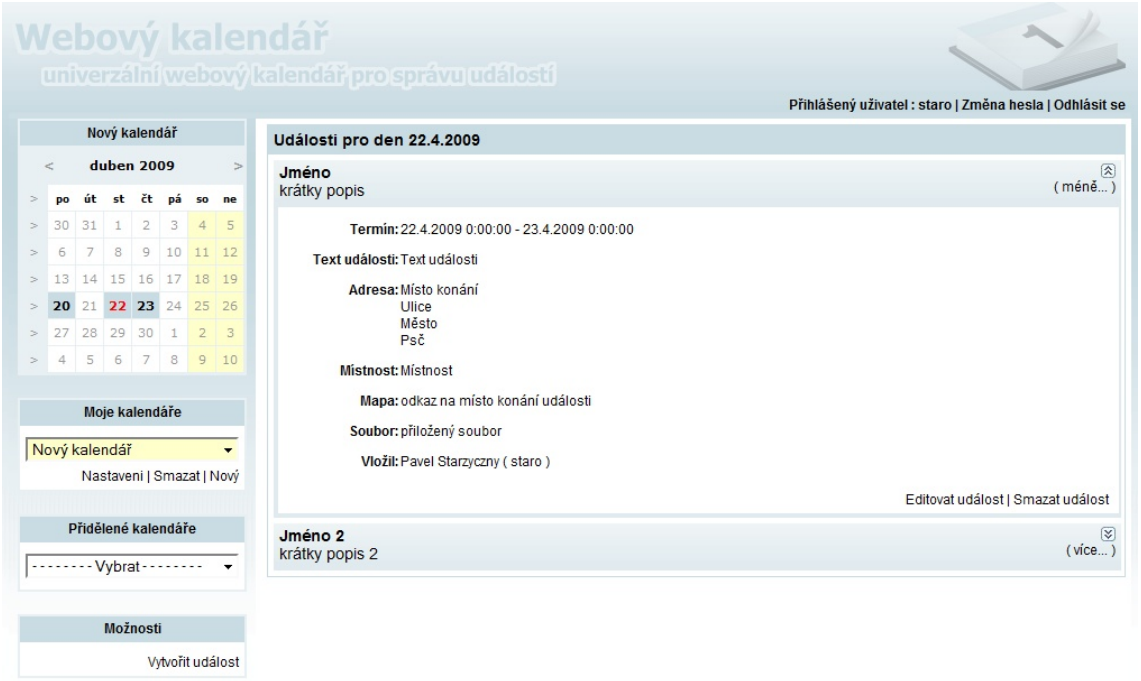
Jak je z obrázku 13 patrné, na levé straně se nachází kalendář s vyznačenými událostmi pro aktuální měsíc. Pod ním je možno vybrat, pro který kalendář se budou události zobrazovat. Dále jsou na výběr buďto kalendáře vlastní, nebo kalendáře jiných uživatelů. S kalendářem můžeme provádět tyto operace:

- nastavení kalendáře
- smazání kalendáře
- vytvoření nového kalendáře

V pravé části se zobrazují události pro vybraný den, týden nebo měsíc. Událostem se zobrazují pouze hlavičky, v nichž je uvedeno jméno události a její krátký popis. Po kliknutí na záhlaví události nebo na šipku v pravé části onoho záhlaví se zobrazí plné detaily události. Po opětovném kliknutí se událost skryje a vytvoří tak místo pro zobrazení události další.

Události můžeme:

- vytvářet
- editovat
- mazat



Obrázek 13: Výsledná administrační část

6.3.2 Prezentační část

Prezentační část, obrázek 14, je tvořena rozšířenou komponentou ASP.NET Kalendář. Při načtení stránky si komponenta stáhne data pro daný kalendář a vyznačí dny s událostmi. Po najetí myší nad vyznačený den se zobrazí na pravé straně (volitelně na levé) okno s událostmi pro daný den. Toto okno připomíná zobrazení událostí v administrační části a funguje na stejném principu. Nejdříve jsou zobrazeny hlavičky událostí se jménem a krátkým popisem. Po najetí myší nad hlavičku události jsou zobrazeny její podrobnosti. Když myš přesuneme mimo událost, detaily se schovají. Toto řešení zobrazování odstranilo problém s horizontálním místem na stránce a s viditelností všech událostí a jejich podrobností v co nejmenším okně. Při zobrazení události jiného dne se okno automaticky aktualizuje. Pro zavření okna událostí slouží křížek v pravém horním rohu.

Obrázek 14: Výsledná prezentační část

6.4 Problémy při vývoji

Během vývoje každé aplikace se vyskytnou problémy, které je nutno vyřešit. Proto v této kapitole budou vybrány některé problémy, jež nastaly pří vývoji této aplikace, a jejich řešení.

6.4.1 Jak zobrazit události v co nejmenším formátu

Hlavním problémem při vývoji prezentační části bylo vyřešit zobrazení událostí na co nejmenším prostoru. Události totiž mohou mít spoustu informací a také dlouhý text. Díky požadavku na zobrazení na co nejmenším prostoru se jako jediná možnost jevilo použití skriptovacího jazyku JavaScript, který umožňuje dynamicky pracovat s obsahem stránky, a proto dokáže zobrazit pouze tu událost, která je požadována. Využití této techniky výrazně snížilo počet dotazů na webovou službu, protože události jsou načítány po měsících a ne po dnech. Zvýšila se i celková interaktivita práce s komponentou.

6.4.2 Jak vzdáleně získat data

Dalším problémem bylo získávání událostí. Jelikož ty jsou umístěny na serveru přístupném z internetu, musela se najít technologie, která zprostředkovává komunikaci mezi dvěma nezávislými systémy. Za úvahu stály technologie AJAX a webové služby. Důležitým faktorem pro výběr byla jednoduchost implementace a zprovoznění. Jelikož přímo k tomuto úkolu jsou navrženy webové služby a jsou integrovány v programu Visual Studio, nemělo smysl se pouštět do technologie AJAX. Toto byla volba správným směrem, jelikož jediné, co bylo potřeba udělat, bylo vystavit službu na internet a využít již naprogramovaný přístup k datům.

6.5 Další možné rozšíření

V budoucím rozšíření by mohly být přidány další funkce jako například exporty dat do XML nebo iCal, jako tomu bylo u aplikace Google Kalendář. Toto rozšíření by umožnilo přesunutí prezentace událostí nejen do vytvořené komponenty a tím pouze na webové stránky, ale i do jiných aplikací, které využívají tyto formáty dat.

Dalším možným rozšířením by mohlo být vytvoření interaktivní grafické plochy pro prezentaci událostí v administrační části. Tento úkon by znamenal pouze záměnu jedné stránky, a jelikož je sama administrační část psána jako třívrstvá aplikace, ostatních funkcí by se tato změna nijak nedotkla.

7 Závěr

Cílem této práce bylo vytvořit aplikaci, která by sloužila jako univerzální webový kalendář v prostředí ASP.NET. Aplikace je rozdělena na administrační část a vlastní serverovou komponentu, která představuje část prezentační. Obě tyto části jsou naprogramovány jako webové aplikace postavené na technologii ASP.NET. Data jsou ukládány prostřednictvím jazyka SQL do databáze běžící na relačním databázovém systému Microsoft SQL Server.

Administrační část je přístupná na stránkách http://kalendar.aspone.cz/. Zde je možno se přihlásit pod vytvořeným zkušebním účtem `jannovak` s heslem `heslo.000`. Prezentační část kalendáře je umístěna na stránkách http://udalosti.aspone.cz/, na kterých se pak zobrazují události z kalendáře výše uvedeného uživatele.

Největší výhodou oproti stávajícím řešením je vytvoření vlastní komponenty prezentující události. Tato komponenta je plně přizpůsobitelná potřebám vývojáře a může být jednoduše přidána do jakéhokoli ASP.NET projektu v programu Visual Studio. Při vyvinutí jen malého úsilí je pak komponenta zprovozněna.

Tato bakalářská práce byla mým prvním webovým projektem v ASP.NET. Díky této práci jsem se naučil základní i pokročilé techniky programování webových aplikací. Tyto zkušenosti jistě využiji ve svých budoucích projektech.

Pavel Starzyczny

8 Reference

[1] MACDONALD, Matthew, SZPUSZTA, Mario. *ASP.NET 3.5 a C# 2008 : tvorba dynamických stránek profesionálně.* Brno : Zoner Press, 2008. 1584 s.

[2] SHARP, John. *Microsoft Visual C# 2005 : krok za krokem.* Brno : Computer Press a.s., 2006. 528 s.

[3] *Wikipedie, otevřená encyklopedie*, <http://cs.wikipedia.org>, 2009.

[4] Prof. Ing. VONDRÁK, Ivo CSc. *Úvod do softwarového inženýrství.* Ostrava : VŠB-TUO, Fakulta elektrotechniky a informatiky, katedra informatiky, 2002. <http://vondrak.cs.vsb.cz/download/Uvod_do_softwaroveho_inzenyrstvi.pdf>

[5] doc. RNDr. ŠARMANOVÁ , Jana CSc. *Teorie zpracování dat* Ostrava : Skriptum VŠB-TUO, Fakulta elektrotechniky a informatiky, 2007.

A Datový slovník

uzivatel							
sloupec	typ	délka	klíč	index	null	IO	komentář
UzivatelID	uniqueidentifier	36	PK	ano	ne	unique	ID uživatele
Login	nvarchar	256	ne	ne	ne		přihlašovací jméno
...							

Tabulka 1: Uživatelé

clenstvi							
sloupec	typ	délka	klíč	index	null	IO	komentář
UzivatelID	uniqueidentifier	36	PK	ano	ne	unique	ID uživatele
Heslo	nvarchar	128	ne	ne	ne		heslo
Email	nvarchar	256	ne	ne	ne		e-mail uživatele
Otazka	nvarchar	256	ne	ne	ne		kontrolní otázka
Odpoved	nvarchar	128	ne	ne	ne		odpověď na otázku
Jmeno	nvarchar	60	ne	ne	ne		jméno uživatele
Firma	nvarchar	50	ne	ne	ano		firma uživatele
ICQ	char	9	ne	ne	ano		ICQ kontakt
Jabber	nvarchar	256	ne	ne	ano		Jabber kontakt
Telefon	nvarchar	14	ne	ne	ano		telefon
...							

Tabulka 2: Členství

kalendar							
sloupec	typ	délka	klíč	index	null	IO	komentář
IDkalendare	uniqueidentifier	36	PK	ano	ne	unique	ID kalendáře
Jmeno	nvarchar	50	ne	ne	ne		jméno kalendáře
Popis	text	max	ne	ne	ano		popis kalendáře
VlastnikID	uniqueidentifier	36	CK	ne	ne		ID vlastníka
Vytvoren	datetime	ne	ne	ne	ne		datum vytvoření
Soukromy	bit	1	ne	ne	ne		soukromý kalendář

Tabulka 3: Kalendáře

udalost							
sloupec	typ	délka	klíč	index	null	IO	komentář
IDudalosti	uniqueidentifier	36	PK	ano	ne	unique	ID události
DatumZacatek	datetime		ne	ne	ne		datum začátku
datumKonec	datetime		ne	ne	ne		datum konce
CasZacatek	datetime		ne	ne	ne		čas začátku
CasKonec	datetime		ne	ne	ne		čas konce
IDkalendar	uniqueidentifier	36	CZ	ne	ne		ID kalendáře
IDuzivatel	uniqueidentifier	36	CZ	ne	ne		ID vkladatele
Jmeno	nvarchar	60	ne	ne	ne		jméno události
Popis	nvarchar	200	ne	ne	ano		popis události
Mesto	nvarchar	50	ne	ne	ano		město konání
Ulice	nvarchar	50	ne	ne	ano		ulice konání
PSC	char	6	ne	ne	ano	čísla	PSČ
Misto	nvarchar	50	ne	ne	ano		místo konání
Mistnost	nvarchar	10	ne	ne	ano		místnost konání
Text	text	max	ne	ne	ano		text události
Mapa	text	max	ne	ne	ano		URL mapy
Soubor	text	max	ne	ne	ano		URL souboru
JmenoPrilohy	nvarchar	256	ne	ne	ano		název přílohy

Tabulka 4: Události

autorizovan_pro_kalendar							
sloupec	typ	délka	klíč	index	null	IO	komentář
IDuzivatele	uniqueidentifier	36	PK/CK	ano	ne	unique	ID uživatele
IDkalendare	uniqueidentifier	36	PK/CK	ano	ne	unique	ID kalendáře
IDvlastnika	uniqueidentifier	36	CK	ne	ne		ID vlastníka

Tabulka 5: Autorizovaní uživatelé